

Modeling and Simulation of 3D Laser Range Scanner With Generic Interface for Robotics Applications

Saso Koceski¹, Natasa Koceska²

^{1,2}Faculty of Computer Science, University Goce Delcev – Stip
st. Krste Misirkov n.10-A, 2000 Stip, Macedonia

*¹saso.koceski@ugd.edu.mk; ²natasa.koceska@ugd.edu.mk

Abstract

The aim of this work is to describe the physics-based simulation model of a custom 3D range laser scanner together with generic interface which offers the possibility for client applications to seamlessly interact with instances of the physical and virtual 3D range scanner. The physics-based virtual model of the 3D range scanner is realized in rigid body dynamics environment provided by Open Dynamics Engine (ODE) library. In order to simulate the stepper motor used in the real scanner, the joints are virtually constrained and motorized with the parameters obtained from real device. For rays modeling the ray-cast method is used. The incidence angle and the surface properties of the object hit by the rays are also taken into the consideration. The sensor noise that represents the uncertainty of the measure is also taken into account. We have also tried to mimic the internal sensor control logic.

In order to verify the developed model and the interface architecture, a simple client application for 3D data acquisition is developed and connected through the same interface to both “physical” and “virtual” device. The results of the evaluation are also reported and discussed.

Keywords

Modeling; Simulation; 3D Range Scanner; Rigid Body Dynamics; Ray-cast Method

Introduction

During the last two decades, the use of robots has gained substantially increasing attraction. Robots participate in meaningful and intelligent interactions with other entities — inanimate objects, human users, or other robots. In order to perform the assigned tasks reliably, a robot first needs to be able to sense its environment accurately. Sensing is the only way through which the robot can assess the status of its environment, and its own status within it.

Moreover, in many practical applications, it is

inevitable for robots to behave in three-dimensional (3D) environments for various tasks, such as navigation, obstacle avoidance, object recognition, manipulation, etc. Therefore, in the field of robotics, there is an increasing need for accurate and low-cost vision sensors for three-dimensional environment perception. Compared to 2D vision systems, 3D visual sensors can provide direct geometric information of the environment. 3D laser range scanners enjoy a rising popularity and are widely used nowadays for these purposes. Many different approaches can be used to build 3D scanning devices. Some of them are based on passive vision: (Narvaez and Ramirez, 2012) for example are presenting 3D scanner based on two web cams; (Lv and Zhiyi, 2011) have constructed an easy to use 3D scanner based on binocular stereo vision using one hand-held line scanner and two cameras. On the other hand, numerous solutions for building 3D scanner based on active vision are reported in the literature. (Früh and Zakhor, 2004) present a system to map cities in 3D using two 2D laser scanners mounted on a truck, one vertically and one horizontally. (Thrun et al., 2000) also use two laser range scanners to obtain 3D models of underground mines. The same setup has been used by (Hähnel et al., 2003) in indoor and outdoor applications. A different approach is presented by (Kohlhepp et al., 2003) which uses 2D laser scanner that points forward and rotates around the front axis. In a similar way, a rotating 2D laser setup is chosen by (Wulf et al., 2004) where the scanner rotates around the vertical axis and is mounted either vertically or pointing upwards. Each of these approaches comes with its own limitations, advantages and elevated costs. Development of an accurate virtual model of a 3D range scanner sensor with off-line programming capabilities (like Easy-Rob and RobotWorks, 2012) together with robot models

will enable fast and cheap testing of different scenarios and algorithms before implementing them in a real environment. Virtual models and simulation tools are often of significant importance in order to reduce development time and avoid damages due control algorithm failure.

Most of the past research in mobile robot simulation has been focused on simulating the structure and kinematics and dynamics of robotic structure, without considering sensors modeling. Recently several robot simulators which are including range scanner models have been developed. Many of these simulators are designed for special robots, only a couple of them offer the possibility of designing own robots. Some of these are restricted to a two dimensional environment (Gerkey et al., 2003, Koenig and Howard, 2004) and plenty of 3D simulators offering 3D range scanner models (Monajjemi et al., 2011; Pincioli et al., 2011; Schmits et al., 2011; Balaguer et al., 2008; Michel, 2004; Gonçalves, 2008; Pereira, 2012). The models of the sensors used in these simulators are too generalized and they typically fail to take into account sensor dynamics and its specific properties, such as the dependency on surface properties, angle of incidence, and uncertainty of the measurement.

In this work, we are presenting a novel approach to create physics-based simulation model of an accurate, low-cost 3D laser scanner developed in our laboratory which is planned to be used in different mobile robotics application. It is based on one 2D range sensor which rotates around a horizontal axis by the means of one stepper motor. The scanner model is realized in rigid body dynamics environment provided by Open Dynamics Engine (ODE) library (Smith, 2012). This library currently supports first order Euler method for the equations' solution. The library uses the Lagrange multipliers' technique and adopts the simplified Coulomb friction model.

In the modeling of the 2D range sensor the uncertainty of the measurement, the dependency of the incidence angle and the surface properties of the object from which it gets reflected, are considered.

In order to achieve more realistic behavior of the simulation model, we have also tried to mimic stepper motor driver board and the internal range sensor control logic. This means to encode range sensor commands into the 2D sensor's protocol in order to feed the data into the very same functions. A novel architecture of a 3D range scanner generic interface which offers the possibility for client applications to

seamlessly interact with instances of the physical and virtual 3D range scanner model is also proposed.

The rest of the paper is organized as follows: first, we explain the design of the physical 3D range scanner, in the next chapter the physics-based virtual model of the scanner is presented, following by a chapter in which the developed application interface is explained. Experimental evaluation of the proposed simulation model as well as the conclusions derived are presented in the last two chapters.

Design of the 3D Range Scanner Prototype

The developed 3D range laser scanner prototype is built on the basis of 2D Time-Of-Flight (TOF) Laser Radar (LADAR) LD-OEM1000 manufactured by SICK (SICK, 2012). The 2D laser range finder is attached to a plate and placed in an aluminum frame, so that it can be rotated (Figure 1). On the top of the plate, berth for camera is foreseen. The rotation axis is horizontal (pitch). In order to have a balanced pivot the construction is made so the weight of the mobile part is along the pivot's axis and the pivot is supported at both ends. Stepper motor (code RS 440-442) from TECO ELECTRO DEVICES for rotation of the movable part is connected on the left side. The motor has a step angle of 1.8° with accuracy of 5%, detent torque of 30mNm and holding torque of 500mNm. In order to enable higher holding torque and higher angular resolution, additional wormshaft gear (series VF 30 P from Bonfiglioli) with transmission ratio of 20 is added.

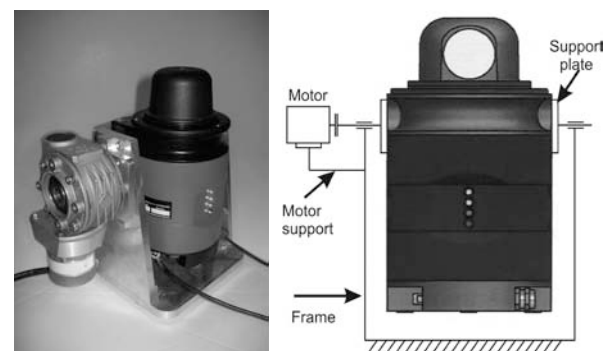


FIG. 1 DEVELOPED PROTOTYPE AND THE SCHEMA OF ITS STRUCTURE

The 3D laser range finder uses only standard computer interfaces. The step motor is directly connected to the parallel port. For the LADAR, CAN data interface was chosen (instead of a RS-232 or RS-422), because it enables high grabbing speed, up to 1 MBit/s, and it can be easily connected via low cost CAN to USB adaptor to an USB port. This permits the developed prototype to be used on every computer

and mobile platforms.

The LADAR measures its environment in two-dimensional polar coordinates. When a measuring beam strikes an object, the position is determined with regard to the distance and direction. The LADAR ranges depend on the scanning object surface properties and angle of incidence and can vary in the interval from 0.5m up to 250m. The scan of 360° (h) x 118.8° (v) is scanned with different horizontal (multiples of 0.125°) and vertical (multiples of 0.09°) resolutions. The scanner head rotates with a frequency of 5 to 20 Hz (programmable).

Development of the 3D Range Scanner Physics-Based Virtual Model

In the creation of the virtual 3D range scanner model the goal is to mimic the physical scanner's behavior as complete as possible and to be able to use the same flow of information like in the physical model. Thus, we have to implement the following parts:

- 2D range sensor and its control unit;
- the stepper motor and its drive board;
- the Encode/Decode component.

The 3D range scanner virtual model is developed in rigid body dynamics environment provided by ODE. This environment allows fairly realistic modeling of physical interactions. The developed simulation model matches closely to the real 3D range scanner in many aspects. All parts that constitute the scanner are modeled as rigid bodies. Each of them has diverse attributes describing its appearance (e. g. color, texture) and physical behavior (e. g. mass, geometrical dimensions, center of mass and moment of inertia). ODE provides several basic geometries: Box, Cylinder, Capped-Cylinder and Sphere. This set has been extended by the so-called ComplexShape. A body of that class has a graphical representation based on a number of geometric primitives and a physical representation based on a set of basic bodies which may approximate the shape.

The LD-OEM1000 sensor has been modeled as a unique body with the same position of the center of mass and the same moment of inertia with respect to the three spatial axes as its real counterpart. Virtual sensor is graphically represented as a complex shape composed of several basic geometries, boxes and cylinders, Figure 2. In its geometrical modeling some approximation is done. For instance, although the

head of the sensor is not exactly cylindrical shaped, we did not consider other alternatives such as meshes, since they are computationally more expensive. The same approach is used for modeling of the other parts of the 3D scanner.

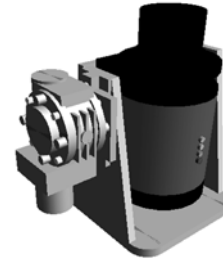


FIG. 2 VIRTUAL 3D RANGE SCANNER MODEL DEVELOPED AS RIGID BODY

For modeling the stepper motor functionality the angular motor in conjunction with ball-socket and hinge ODE joints are used. By attaching these joints between the frame and the support plate, angular velocity can be controlled on up to three axes, allowing torque motors and stops to be set for rotation about those axes. In our case the movements are limited only around the horizontal axis.

The rotary motor operates by applying a torque to the hinge joint in the dynamics simulator. The amount of torque T applied to the hinge axis is based on the configuration of the motor and the voltage applied to the motor, using equation 1.

$$T = Gk_t \frac{(V - G\omega / k_v)}{R} \quad (1)$$

V is the amount of voltage applied to the motor, k_t is the torque constant of the motor, k_v is the back emf constant, R is the internal motor resistance, G is the gear head ratio of motor turns to output shaft turns (if a gear head is attached), ω is the current angular velocity of the output shaft. Once the torque is computed, this information is passed to the hinge joint in ODE for correct dynamics simulation during the next simulation step.

In the simulator the speeds are changed by setting the desired angular speed and maximum torque of the motor connecting the plate with the frame. The desired speed is reached in the shortest time possible, i.e. by applying the maximum torque, and this speed maintained afterward.

ODE's ray collision primitives are used to simulate the laser beams. We define a ray that simulates the beam with the following characteristics: the start position, the ray direction, the minimum threshold (D_{min}),

corresponding to the minimum measurable distance of the sensor, and the maximum threshold (D_{\max}), corresponding to the maximum measurable distance of the sensor. In our case, by the experimental tests we have obtained the value of $D_{\min} = 0.9\text{m}$ for the minimal threshold for which the sensor is measuring correctly, and we have set the value of $D_{\max} = 30\text{m}$ (which satisfies the needs of our applications).

Rays are generated from the ray source. First intersection of these rays with the objects in the environment is returned as the distance information. In the simulation, instead of time of flight, direct distance is obtained using low level functions. The principle involving the range scan is also similar. At each simulation step only a single planar scan is made with the scanner. Thus, 3D scans need more simulation steps to move the motor to required positions. In the modeling of the range sensor the dependency of the incidence angle and the surface properties of the object from which the rays get reflected, should be also considered.

In general, reflections can be modeled as combination of specular and diffuse reflections. Perfect specular reflections (as would occur on a perfect mirror) obey the specular reflection law which states that the angle of incidence equals the angle of reflection. The beam is not dilated in the process and continues not displaying any beam divergence. In contrast, diffuse reflection spreads the incoming light over the entire hemisphere surrounding the surface of incidence.

In the implemented model, the reflection properties of the objects in the environment are computed following Lambert's cosine law, which models diffuse reflections. Specular reflections are neglected since situations in which perfect specular reflections return to the sensor in the setup described above are rare.

It is assumed that the emitter generates a laser beam of intensity I_0 which does not display any beam divergence and that at the target object it is scattered homogeneously in all backward directions i.e. the reflected flux is distributed equally over a hemisphere (Lambertian surface). The detector is co-located with the emitter, facing the same direction and is assumed to have a circular receptor surface of diameter d .

Hence the reflected light can be thought to originate from a Lambertian point source located at the point of incidence. An illustration of the sensor model is shown in Figure 3.

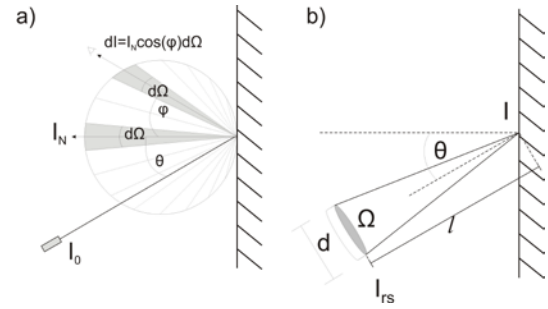


FIG. 3 SENSOR MODEL

Lambert's law states that the light intensity dI emitted in an infinitesimally small solid angle $d\Omega$ at angle θ from the surface normal is

$$dI = I_N \cos(\theta) d\Omega \quad (2)$$

with I_N the luminous intensity which corresponds to the intensity emitted in direction of the surface normal at the point of incidence of the laser beam. I_N is the maximal intensity reflected in any direction by the Lambertian point source. The total light intensity reflected at the point of incidence is

$$I = \int_0^{\pi/2} I_N \cos(\theta) 2\pi \sin(\theta) d\theta \quad (3)$$

In equation 2, the solid angle $d\Omega$ has been replaced by the solid angle which corresponds to a spherical segment (zone) of aperture angle $d\theta$. The integration from 0 to $\pi/2$ results in the entire hemisphere of reflectance being considered. The total intensity of the reflected light is related to the emitted intensity by

$$I = rI_0 \quad (4)$$

with r being the coefficient of reflection ($0 \leq r \leq 1$), the only modeled optical material property of the environment. Solving equation 2 for I_N yields

$$I_N = \frac{I}{\pi} = \frac{rI_0}{\pi} \quad (5)$$

The light intensity I_{rs} received at the sensor depends on the solid angle Ω covered by the receptive surface in relation to the surface of a hemisphere of radius l - which is the total surface into which the light is emitted:

$$\Omega(l, d) = \frac{d^2 \frac{\pi}{4}}{\frac{1}{2} 4\pi l^2} = \frac{d^2}{8l^2} \quad (6)$$

In equation 5, the surface of the sensor is approximated by a (planar) circular surface. Combining Lambert's cosine law with equation 4 and equation 5 yields the light intensity received at the

sensor:

$$I_{rs} = I_N \cos(\theta) \Omega = \frac{rI_0}{\pi} \cos(\theta) \frac{d^2}{8l^2} \quad (7)$$

When generating the sensor output, it is assumed that if sufficient light intensity is sensed, a correct distance value can be determined. If the intensity drops below a threshold, no output is generated.

The sensor response model described above has been validated using data contained in the technical specifications provided by the sensor manufacturer (SICK, 2012), which relates minimum required reflectance values to target range, supposing that a full spot strike on the object. A good agreement between the model and the technical specifications data are found for range values higher than 0.5 meters using a value of $I_{r\min} = 0.00016$ [photons/(s•cm²•sr)] (specifying $I_0=1$ [photons/(s•cm²•sr)] and $d=4$ cm). The fact that only one free-parameter ($I_{r\min}$) is required to fit data for four different values of visibility indicates that the physical basis of the model is reasonable at range values higher than 0.5 meters.

We also modeled the sensor noise that represents the uncertainty on the measurement. We approximate this error by a Gaussian law. The noise is then defined by:

$$N = \sigma * G + m \quad (8)$$

Where σ is the standard deviation, m the mean of the Gaussian law, and G a Gaussian variable that can be computed by:

$$G = \sqrt{-2 \cdot \ln(R_1)} \cdot \cos(2\pi R_2) \quad (9)$$

where R_1 and R_2 are two variables uniform in law on [0,1].

The implementation of the LD-OEM control unit is based on the sensor's host protocol. The host protocol provides a set of commands to control the LD-OEM. This set of commands is divided into different service groups. When the LD-OEM has received a host protocol command, it answers with a certain response. The LD-OEM responds when the service has been processed. Generally, this takes less than one second. An exception is the service TRANS_ROTATE, which can take several seconds.

In order to determine the kind of service of a host request the service codes are used. It can be seen as the command. Its data format is WORD. The most significant bit defines whether the code is a request or a response. For a request the most significant bit is set to zero and for a response it is set to one. Usually the

LD-OEM does not request any data.

Consequently, requests are sent by the host whereas responses are sent by the LD-OEM. The remaining of the high byte determines the service group and the low byte defines the service number (Table 1).

TABLE 1 SERVICE CODE STRUCTURE

Bit 15	Bit 14...8	Bit 7...0
Response bit	Service group	Service number
	0 ... 127	0 ... 255

LD-OEM provides the following services:

- Status Services
- Configuration Services
- Measurement Services
- Working Services
- Maintenance Services
- Routing Services
- File Services
- Monitor Services
- Application Services
- Adjust Services
- Special Services

The control unit's encoder/decoder component which is responsible to translate the input and output messages, in the current implementation is capable to process the first four groups of services. A service request sent with missing or invalid parameters leads to a response, which indicates in the return value, that the request was invalid. An invalid command request is answered by a SERVICE_FAILURE (FF00h).

The stepper motor drive board is implemented like a separate unit which continuously read the LPT port status. It accepts basically three types of signals related to the movement of the stepper motor:

- Direction (which corresponds to the LPT DATA0 line) - according to the level of this signal, the stepper motor will rotate clockwise or antidclockwise.
- Clock (which corresponds to the LPT DATA1 line) – is the impulse which commands the execution of one motor step. The frequency of the squared impulses will determine the velocity of the stepper

motor.

- Mode (which corresponds to the LPT DATA2 line) – when this signal is maintained on low level the stepper motor works in one-Phase-on mode performing 200 steps per cycle. Otherwise, the motor is working in Half-step mode performing 400 steps per cycle.

3D Range Scanner Interface Architecture

The contextual diagram of the 3D range scanner's interface architecture is shown in Figure 3. The architecture provides the possibility for different client applications to be connected to the 3D range scanner instances, physical or virtual, through a single interface. This implies that one part of the interface should be able to handle different application requests, and thus it needs to be able to manage various implementation issues. The interface is composed of several functional modules. Each one needs to address specific issues:

- message Encoders/Decoders, which also check and report errors,
- external and internal message handlers that will act on predefined client and sensor messages,
- communication components.

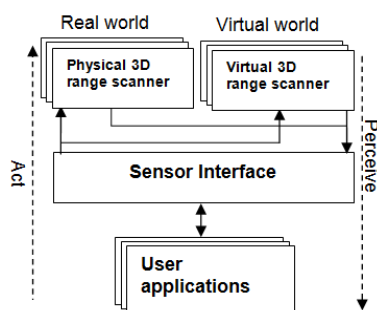


FIG. 4 CONTEXT DIAGRAM OF THE SYSTEM INCLUDING THE 3D RANGE SCANNER'S INTERFACE ARCHITECTURE

The top level 1 data-flow diagram of the system is shown on Figure 5.

Message Encoders/Decoders

The Encoders/Decoders are responsible for transforming messages in the proper format, between the client application and the Front-end Message Handlers on one hand and between Back-End Message Handlers and different 3D range scanner instances (physical or virtual) on the other. One or more are needed by client applications and one is normally needed per device. Encoders/Decoders are

also responsible for error handling and error reporting.

Front-End and Back-End Message

The Front-end Message Handler (FEMH) is responsible for handling incoming requests from the user and if necessary, re-directs them to a Back-End Message Handler (BEMH). Depending on the type of the request received, the FEMH instance decides whether to read already obtained sensor data or ask the BEMH to process the request i.e. to pass the request on to the sensor. FEMH is application specific i.e. handles requests from and responds to a specific user application. The BEMH routes requests to the sensor. Each sensor physical or virtual is handled by a BEMH instance. Thus, there are as many BEMH instances as sensors attached to the interface. When the BEMH instance gets the response back from the sensor, it redirects the response to the proper FEMH instance which subsequently responds to the user. On creation of the handler instances the unique identifiers are assigned to them and registered by the interface and are preserved during the current working session.

Communication Units

The aim of the communication units in the system is to enable message delivery via a physical device between the sender's encoder and recipient's decoder. Communication units have very important role in mobile robotics applications where usually sensor's and user's logic reside on different locations. Various communication components exist in the system. In our case the front-end communication unit is based on TCP/IP protocol, while the back-end unit is based on CAN communication protocol for the range sensor and another simple protocol for the stepping motor, based on the three types of signals as described before. In both cases an implementation is necessary in the Encode/Decode components. Some specific issues about the CAN protocol are reported in following.

Experimental Work

In order to verify both the 3D range scanner model and the proposed interface architecture, a simple client application that can control the 3D range scanner and can collect 3D data from the environment is developed. The simulator, scanner interface modules and client application are implemented using C and C++ programming languages and ODE libraries on PC with AMD Athlon 64 Processor on 2.4GHz with 1GB of RAM, NVIDIA GeForce FX5500 with 256MB

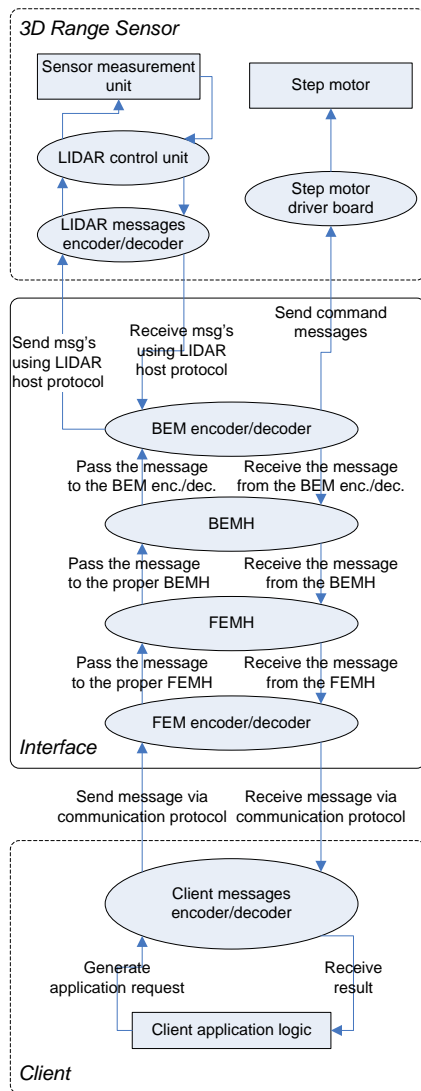


FIG. 5 FIRST LEVEL DATA-FLOW DIAGRAM OF THE MAIN SYSTEM COMPONENTS

memory and Windows XP operating system. The visualization is done using OpenGL. The application is connected first to the physical and after that virtual 3D range scanner instance.

A simple scene (Figure 6) in the real world is created and scanned. The scanner was placed on a wheeled wooden cart at a 4180mm distance from a large wall. Several simple shaped objects were placed in front of the wall and scanned with horizontal apex angle of 20° , vertical apex of 21.6° , the horizontal resolution of 0.25° and the vertical resolution of 0.18° .

Simulation environment and all objects in it are defined using XML files, which are parsed to construct the virtual world. This reduces the task of loading a scene to a single line of code. This allows the creation of much more complex environments and models easily. XML files can also be easily extended to add additional user parameters. Virtual scanner is

configured with the same set of parameters. In both cases the same sequence of commands is send: INITIALIZE – to initialize the 2D sensor, MOVEMOTOR (with attributes which indicate the number of steps and the direction) – to move the 2D sensor in the low vertical apex position, ROTATE – start the rotation of the 2D sensor head, MEASURE – start the acquisition process, GET2DPROFILE – acquire one 2D profile, MOVEMOTOR (with attributes which indicate one step forward in the scanning direction), IDLE – to exit the acquisition mode, MOVEMOTOR - to return the LADAR in the initial position.



FIG. 6 REAL 3D RANGE SCANNER FACING AGAINST A CARTON CYLINDRICAL OBJECT IN THE UNIVERSITY CORRIDOR

The same scene is also virtually modeled (Figure 7) using ODE.

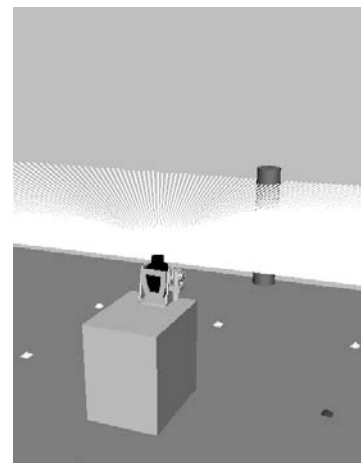


FIG. 7 VIRTUAL 3D RANGE SCANNER MODEL FACING AGAINST A VIRTUAL CYLINDRICAL OBJECT

The typical flow of information trough the interface, during the acquisition process is the following:

- Client application sends a request to the desired 3D range scanner using some protocol (e.g. TCP/IP)

or direct connection.

- The interface extracts and identifies the client from the request and calls the proper FEMH instance.
- Dependent on the request, the FEMH instance decides if it is able to get the information immediately from the scanner's BEMH instance or if it needs to ask the scanner's BEMH instance to pass the request on to the 3D range scanner.
- When the 3D range sensor BEMH instance gets the response back from the 3D range scanner, it redirects the response to the proper FEMH instance which in turn responds to the client.

On Figure 8 the point cloud obtained with the scanning of cylindrical object in the real environment is presented. Local errors, which can be referred to as noise, stayed within the range -15 to 15 mm. In Figure 9, the scan output of the virtual cylindrical object is given. In the virtual model a Gaussian distributed noise with a zero mean and 0.15 cm of standard deviation was added to each distance measurement. It can be observed that the scan output looks very

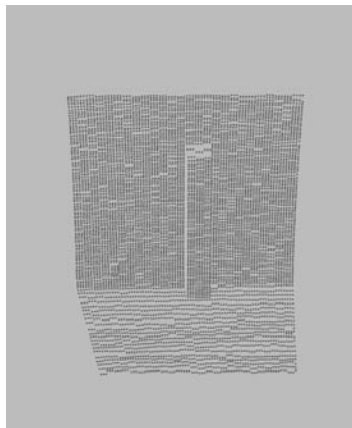


FIG. 8 REAL SCANNER READINGS IN THE FORM OF POINT CLOUD

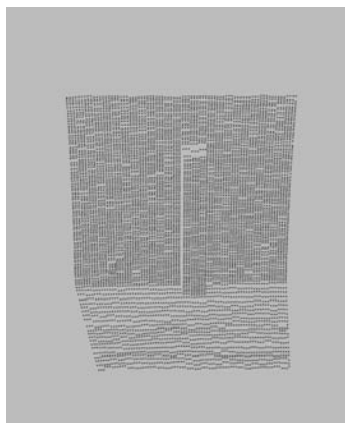


FIG. 9 SIMULATED SCANNER READINGS IN THE FORM OF POINT CLOUD

realistic. The cylinder object was fitted in both cases. In the case of the real scan, the fitted cylinder is with diameter of 196mm and in the case of virtual scan it has diameter of 197mm. In both cases the length of the cylinder was 1m. These results are comparable with the dimensions of the real cylindrical object which has diameter of 195mm and length of 0.998m. One can also observe the absence of mixed pixels in the simulated scan at the top of the cylinder.

Conclusions and Future Work

This paper proposed a physics-based simulation model of a custom 3D range laser scanner. Experimental work was carried out in order to compare the scanner model against the real one. The results show good correspondence between the two measurements. The sensor and the simulation environment with all their properties are parametrically defined in XML format which enable easy extension of the concept on other 3D scanner devices. With the developed scanner interface on the other hand seamless integration with physical and virtual sensors is enabled. This enables the client applications to have a single point of contact with both types of sensors which is important in our future development.

However, the developed model currently is not considering the reflectance of the scanned objects and environment conditions (for e.g. temperature and humidity which are influencing the measurements). So, our future research will be focused on this topic. Building of entire and flexible mobile robot simulator and other types of proximity sensors is also foreseen.

REFERENCES

- Balaguer, Benjamin, and Stefano Carpin. "Where Am I? A Simulated GPS Sensor for Outdoor Robotic Applications." *Simulation, Modeling, and Programming for Autonomous Robots* (2008): 222-233.
- Easy-Rob, 3D Robot Simulation Tool, website <http://www.easy-rob.de/> (2012)
- Früh, Christian and Zakhor Avidoh. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60:5-24, October 2004.
- Gerkey, Brian P, Richard T Vaughan, and Andrew Howard. 2003. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Advanced Robotics*, ed. U

- Nunes, A T de Aalmeida, A K Bejczy, K Kosuge, and J A T Macgado, 26:317-323
- Gonçalves, José, José Lima, Hélder Oliveira, and Paulo Costa. "Sensor and actuator modeling of a realistic wheeled mobile robot simulator." In *Emerging Technologies and Factory Automation*, 2008. ETFA 2008. IEEE International Conference on, pp. 980-985. IEEE, 2008.
- Hähnel, Dirk, Wolfram Burgard, and Sebastian Thrun. "Learning compact 3D models of indoor and outdoor environments with a mobile robot." *Robotics and Autonomous Systems* 44, no. 1 (2003): 15-27.
- Koenig, N, and A Howard. 2004. Design and use paradigms for gazebo, an open-source multi-robot simulator. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems IROS IEEE Cat No04CH37566 3: 2149-2154
- Kohlhepp, Peter, Marcus Walther, and Peter Steinhaus. "Schritthalte 3D-kartierung und lokalisierung für mobile Inspektionsroboter." *Proceedings of the Autonome Mobile Systeme* 18 (2003).
- Lv, Zhihua, and Zhiyi Zhang. "Build 3D laser scanner based on binocular stereo vision." In *Intelligent Computation Technology and Automation (ICICTA)*, 2011 International Conf. on, vol. 1, pp. 600-603. IEEE, 2011.
- Michel, Olivier. 2004. WebotsTM: Professional Mobile Robot Simulation. *International Journal of Advanced Robotic Systems* 1, no. 1: 39-42.
- Monajjemi, Valiollah, Ali Koochakzadeh, and Saeed Ghidary. "grSim-RoboCup Small Size Robot Soccer Simulator." *RoboCup 2011: Robot Soccer World Cup XV* (2012): 450-460.
- Narvaez, Astrid, and Esmitt Ramirez. "A Simple 3D Scanner Based On Passive Vision for Geometry Reconstruction." *Latin America Transactions, IEEE (Revista IEEE America Latina)* 10, no. 5 (2012): 2125-2131.
- Pereira, José LF, and Rosaldo JF Rossetti. "An integrated architecture for autonomous vehicles simulation." In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pp. 286-292. ACM, 2012.
- Pincioli, Carlo, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews et al. "ARGoS: a modular, multi-engine simulator for heterogeneous swarm robotics." In *Intelligent Robots and Systems (IROS)*, 2011 IEEE/RSJ International Conference on, pp. 5027-5034. IEEE, 2011.
- RobotWorks — A Robotics Interface and Trajectory Generator for SolidWorks, website <http://www.robotworks-eu.com/> (2008)
- Schmits, Tijn, and Arnoud Visser. "An omnidirectional camera simulation for the USARSim World." *RoboCup 2008: Robot Soccer World Cup XII* (2009): 296-307.
- SICK, <http://www.sick.com> (2012)
- Smith, R.: Open Dynamics Engine - ODE <http://www.ode.org> (2012)
- Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping." In *Robotics and Automation*, 2000. *Proceedings. ICRA'00. IEEE International Conference on*, vol. 1, pp. 321-328. IEEE, 2000.
- Wulf, Oliver, Kai O. Arras, Henrik I. Christensen, and Bernardo Wagner. "2D mapping of cluttered indoor environments by means of 3D perception." In *Robotics and Automation*, 2004. *Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 4, pp. 4204-4209. IEEE, 2004.